

Macintosh® Programmer's Workshop Toolbox Interfaces and Libraries

Version 7.0

M0615LL/B



030-8016-A

Apple Computer, Inc.
20525 Mariani Avenue, M/S 33-G
Cupertino, CA 95014
(408) 996-1010
TLX 171-576

To reorder products, please call:
1-800-282-2732 (in the United States)
1-800-637-0029 (in Canada)
1-408-562-3910 (International)

MPW 3.2 Libraries/Interfaces
Release Notes

Libraries

Changes to the libraries since MPW 3.0 include the elimination of CInterface.o and CRuntime.o, their functionality having been merged into Interface.o and Runtime.o. The libraries have been resegmented so as to reduce the number of modules in segment "main". The MPW 3.2 C libraries conform to the current ANSI C standard. A number of modifications and additions have been made to the set of library functions, including new C-like string functions for Pascal strings and a C function for setting the type and creator on any file. A new library provides for the creating of applications that use a simple text I/O window to display and read console I/O (SIOW). For details, see the MPW 3.2 Simple Input/Output Window Release Notes.

Linking Requirements

The library CLib881.o has a new positioning requirement as a consequence of the elimination of CRuntime.o. CLib881.o must now not only precede all members of {CLibraries} in the link sequence but must also precede the library Runtime.o.

Note that XCMDs for Hypercard require linking into one segment. Because of the new segmentation of the library, this will no longer happen automatically. This requirement can be met by putting the option `-sg main` on the link command line to force the library routines and your code into a single segment.

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

Segmentation

The libraries have been resegmented, moving many of the modules out of the segment "Main". The goal of changing the segmentation is to minimize the number of modules in the segment "Main". This way, if the user needs the space in "Main" for his application or tool, he has it. If he is more concerned about speed, the user can then manipulate his makefile to contain whatever he needs in his "Main" segment.

- Segment layout

LIBRARY : Runtime.o

Segment	Contents
MAIN	Initialization and exit procedures, 32 bit math operations
%A5Init	A5 initialization code, _DataInit, Module #0001
INTENV	Low level IOPort functions and I/O which directly calls those functions - done for pc relative offsets
SADEV	Device drivers (Stand Alone Devices)

LIBRARY : Paslib.o

Segment	Contents
MAIN	Compiler initializations (the functions call Runtime.o)
INTENV	I/O which directly calls segment INTENV in Runtime.o (may change in the future)
STDIO	Pascal language I/O
PASLIB	Heap, strings and math functions

LIBRARY : StdCLib.o

Segment	Contents
STDIO	C language I/O
STDCLIB	All ANSI functions (except I/O)

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

LIBRARY : Stubs.o

Segment	Contents
SADEV	Device drivers (stubs out the driver functions)

LIBRARY : DRVRRuntime.o

Segment	Contents
MAIN	Driver initialization and exit procedures

Changes to C Library

ANSI C

The MPW 3.2 C libraries are largely in conformance with the current ANSI C standard (X3.159—1989).

File Types

If you create a new file and do not specify that it is a binary file by including `'b'` as part of the open mode of an `fopen()` call or `F_BINARY` as part of the mode for an `open()` call, the file that is created will be of type TEXT. Except for marking a file as a TEXT file at its creation, there is no difference in the way that MPW C handles text and binary files internally.

There is also a new function which can be used to set the creator and file type of any file. The function is:

```
void fsetfileinfo (char *filename, OSType newcreator,
                                   OSType newtype);
```

cfree()

Because `cfree()` is not part of the ANSI standard this function has not been supported since the release of MPW 3.0. The code for `cfree()` has now been removed from the libraries. If you have any existing code which uses this function, you can create your own macro or function to replace it. Since MPW C has always ignored the second two arguments of `cfree()`, the function call `free(p)` is equivalent to `cfree(p, int1, int2)`.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

dup()

Because `dup()` is not part of the ANSI standard and because its functionality is duplicated in the `fcntl()` function, the `dup()` function is no longer supported and will be removed from a future version of `StdClib.o` in the MPW C Libraries. The function call `fcntl(filedes, F_DUPFD, 0)` is exactly equivalent to the call `dup(filedes)`. Existing code should either be changed to call `fcntl()` directly or a macro or local definition of `dup()` should be provided by the user.

fputs()

This function will now set *errno* when applied to a read-only file.

Performance Improvements

The speed of the following functions has been improved:

memchr()	strchr()	strncmp()
memcmp()	strcmp()	strncpy()
memcpy()	strcpy()	strpbrk()
memmove()	strcspn()	strrchr()
memset()	strlen()	strspn()
strcat()	strncat()	strstr()

Previously Undocumented Limitation

`malloc` is currently limited to a maximum size of about 8 MB per call. Size parameters of value 2^{23} or greater result in a null pointer being returned.

Pascal Implementation of C String Functions

Pascal string functions that are equivalent to the standard set of C string functions have been implemented. Two new interface files enable the calling of these functions from either C or Pascal programs. From Pascal programs, PLStringFuncs.p and Types.p are required. From C programs, PLStringFuncs.h is required.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

◆ *Note:* Some of these functions may be redundant in C, Pascal, or both, but are provided for consistency with the standard C functions and to make the functionality directly available in all languages. In particular, the function PLstrlen is redundant since Pascal provides the standard function Len() for this purpose and C can directly manipulate the first byte of a Pascal string which contains the length of the string. Similarly, the function PLstrcmp is redundant in Pascal because the relational operators such as "=" can be used to compare strings. PLstrcmp is useful in C, however, since there is no direct way to compare Pascal strings in C.

△ **Important** Many of these routines assume that the strings have been declared as String[255] (in Pascal) or are stored in a buffer of at least 256 bytes (in C). Smaller string buffers can be used, but care must be taken when performing operations which may extend the length of the string. These operations (such as PLstrncat) will overwrite memory if the string is extended beyond the declared length. △

The function descriptions follow:

function PLstrcmp (string1, string2 : Str255) : integer;

This function compares string1 and string2. It returns an integer greater than, equal to, or less than zero, depending on whether string1 is greater than, equal to, or less than string2.

function PLstrncmp (string1, string2 : Str255; n : integer) : integer;

This function compares not more than n characters from string1 to string2. It returns an integer greater than, equal to, or less than zero, depending on whether the first n characters of string1 is greater than, equal to, or less than the corresponding portion of string2.

function PLstrcpy (string1, string2 : Str255) : StringPtr;

This function copies the contents of string2 into string1. It returns a pointer to string1.

function PLstrncpy (string1, string2 : Str255; n : integer) : StringPtr;

This function copies not more than n characters from string2 to string1. It returns a pointer to string1. If string2 is shorter than n characters, null characters (Chr(0)) are written to string1 until n characters have been written.

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

function PLstrcat (string1, string2 : Str255) : StringPtr;

This function appends characters from string2 to string1, starting with the first character of string2, and stopping when string2 is exhausted or when the expanded string1 attains a length of 255 characters, whichever happens first. It returns a pointer to string1.

function PLstrncat (string1, string2 : Str255; n : integer) : StringPtr;

This function appends characters from string2 to string1, starting with the first character of string2, and stopping when n characters have been copied, when string2 is exhausted, or when the expanded string1 attains a length of 255 characters, whichever happens first. It returns a pointer to string1.

function PLstrchr (string1: Str255; c : Char) : Ptr;

This function locates the first occurrence of c in string1. It returns a pointer to the located char, or NULL if c was not found.

function PLstrrchr (string1: Str255; c : Char) : Ptr;

This function locates the last occurrence of c in string1. It returns a pointer to the located char, or NULL if c was not found.

function PLstrpbrk (string1, string2 : Str255) : Ptr;

This function locates the first occurrence in string1 of any of the characters in string2. It returns a pointer to the character, or NULL if no character from string2 occurs in string1.

function PLstrspn (string1, string2 : Str255) : integer;

This function computes the length of the first segment in string1 that consists entirely of characters from string2. It returns the number of characters in that segment.

function PLstrstr (string1, string2 : Str255) : Ptr;

This function locates the first occurrence of string2 in string1. It returns a pointer to the located string, or NULL if the string was not found.

function PLstrlen (string1 : Str255): integer;

This function returns the length of string1.

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

function PLpos (SubStr, MainStr : Str255): integer;

This function locates the first occurrence of SubStr in MainStr. It returns the position in mainStr at which SubStr was found.

Changes to ToolLibs.o

ErrMgr

A fix has been added to treat error -42 (too many open files) as a special case. This was done because if this error actually did occur, then the ErrMgr would normally not be able to report it because it would not be able to open its own error message file!

Disassembler

Enhanced to support the MC68040 extensions.

Changes to Interface.o

The glue has been systematically revised to shorten the routines or to convert them to in-line code in the Interfaces.

The 64K ROMs are no longer supported.

Folder Manager glue now exists for System 6 support.

In PPCToolbox, the glue for `PromptForUser` has been removed.

In Gestalt, the "mach" glue has been updated so that it works on Mac Plus and Mac SE.

△ **Important** Do not use `FSOpen` to open drivers. The glue first makes a call to `_OpenDF`, which will only open files. It then calls `_Open` if `_OpenDF` is not implemented. Use the call `OpenDriver` instead.

`GetHandleSize` and `GetPtrSize` have been changed so that they return a size of zero if an error has occurred. You should always check `MemError` to find out whether an error occurred. △

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

Miscellaneous

HyperXLib.o is the HyperCard 2.0 final library.

Known Outstanding Bugs

The items listed below are bugs that existed in MPW 3.0 or MPW 3.1. The list does not include bugs that first appeared in Alpha or Beta versions of MPW 3.2.

ObjLib.o

ObjLib.o does not yet support 32-Bit Everything, so tools or applications linked with this library cannot use `-model far`.

StdClib.o

The error numbers returned in `errno` by the `fcntl` function are not in agreement with the documentation. If the `filedes` parameter is not an open file, `fcntl` returns `EINVAL` instead of `EBADF`. Similarly, if the `arg` parameter to `fcntl` is negative or larger than the largest allowable file descriptor, `fcntl` returns `EBADF` instead of `EINVAL`. It has not yet been determined whether this will be resolved by a code change or a documentation change.

The values returned by the functions `islower`, `isalnum`, `isspace`, `isupper`, `isalpha`, `isdigit`, and `isprint` are randomly incorrect for Macintosh special characters (characters whose representation as type `signed char` is negative).

Bug Fixes

The items listed below are bugs that existed in MPW 3.0 or MPW 3.1. The list does not include bugs that first appeared in Alpha or Beta versions of MPW 3.2.

- Under certain circumstances, when given a "leaf" (pathless) filename, the functions `fopen` and `open` were found to be using the Macintosh "Poor man's search path," i.e., if they failed to find the specified file in the current directory, they might search certain other paths, notably the system folder, for a file of the same name. This has been (and still is) an option, but it no longer occurs unintentionally.
- `GetIndString` no longer returns rubbish values for 0 and negative numbers (large numbers).

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

- `StartSound` now works.
- `Create` no longer does a `SetFileInfo` with random data when `GetFileInfo` fails. It now correctly returns an error code.

Other Useful Information

`SysBreak`, `SysBreakStr`, and `SysBreakFunc` are three new routines defined to make debugging with SADE and Macsbug easier. In the past, as a debugging aid, the routines `Debugger` and `DebugStr` were provided which allowed programmers to force the application to drop into their favorite debugger (MacsBug, TMON, The Debugger, etc.). When SADE was developed, it was decided that it would be advantageous to allow SADE to be used in conjunction with a low level debugger. Thus SADE does not handle calls from `Debugger` or `DebugStr`; they will only be intercepted by your low-level debugger. The routines `SysBreak`, `SysBreakStr`, and `SysBreakFunc` are designed to trigger the "highest-level" debugger available. If SADE is running, then SADE will handle the call. If SADE is not running then the low-level debugger will handle the call. A small problem arises however when running without MultiFinder and without any debugger installed. When either of these routines is called, the System Error handler fails to find a 'DLOG' resource and the Dialog items are displayed on the desktop.

Of course, shipped applications have all of their debugging code removed, and the end user will never encounter this problem. Well almost: there are certain error conditions where the libraries will call one of these routines (heap corrupt, etc.). This situation will be addressed in a future release of the libraries and interfaces.

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

Interfaces

△ **Important** Safety tip: do not throw away your current interfaces until you have read these notes and made sure that you can still build your products. △

△ **Important** C programmers: beware the use of `sizeof` for doing pointer arithmetic. If the size of a structure changes, you may wind up with code that compiles properly but gives you the wrong result. For example, zero-length arrays at the end of some records are used to indicate that variable-length data follows. However, C++ does not allow zero-length arrays, so the best that can be done is to make them one byte long. If you use `sizeof` to find the start of the variable-length data, you are now pointing two bytes into the data (one byte for the array, one byte for padding).

Use instead the `offsetof` macro (defined in StdDef.h) to locate the start of the variable size array. It will not be affected by the array size. △

The file called {MPW}Interfaces:Interfaces.Dict is a cumulative dictionary file that can be used in conjunction with the Canon tool to convert your sources to correspond to the changed spellings in the interfaces. Instructions for its use are in the comments to be found at the start of the file.

Extensive additions have been made to the Interfaces in support of System Software 7.0. Attention is also directed to the changes shown below in existing calls.

General Interface Changes

The conditional flags `SystemSixOrLater` and `SystemSevenOrLater` have been defined for the use of developers who wish to target their software for running only under specific OS versions. This enables them to reduce their code size.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

Changes to C Interfaces Only

ANSI C headers

None of the ANSI header files contain a *#include* of any other header file.

#defines for integers

With the exception of the ANSI header files, the use of `#define` to define symbols as integers has been largely eliminated. In most of the interface files, such instances have been replaced by `enum` declarations. E.g.,

```
#define A 3
#define B 4
#define C 1
...
```

has been replaced by

```
enum {A = 3, B = 4, C = 1, ...};
```

This has been batched as twenty such definitions per enum.

String Pointer Parameters

The definition `typedef const unsigned char *ConstStr255Param` has been introduced in types.h for compilers that support `const`. It is intended to replace the use of `const Str255`. For consistency, similar declarations have been included for `ConstStr63Param`, `ConstStr32Param`, `ConstStr31Param`, `ConstStr27Param`, and `ConstStr15Param`.

Generic Pointers

In function declarations, all instances of `Ptr` as an argument type have been changed to `void *`. This removes the restriction that parameters represented by pointers had to be of type `char *`. It is intended in the near future to make the equivalent change for Pascal by using `UNIV Ptr`.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

C++

The `__safe_link` compile time variable was changed to `__cplusplus`, and the conditional code around typedefs of the form

```
#ifndef __cplusplus
      typedef struct Point Point;
#endif
```

was removed (it is no longer required for C++ compilation).

Aliases.h

In the call `MatchAlias`, the parameter `aliasList` has had its type changed from `FSSpecPtr` to `FSSpecArrayPtr`.

AppleEvents.h

Function prototypes no longer pass an AppleEvent structure. Instead, they now pass a pointer to the structure.

AppleTalk.h

`const AddrBlock *netAddr` in the `BuildDDPwds` call has been changed to `const AddrBlock netAddr`.

The Socket parameter in the calls `DDPCloseSocket`, `DDPOpenSocket`, `ATPCloseSocket`, and `ATPOpenSocket` are now of type `short`. They were incorrectly changed to `char` during some earlier alpha and beta releases of MPW 3.2.

The parameter `addrRcvd` in `ATPOpenSocket` has been changed from `const AddrBlock *` to `AddrBlock`.

Balloons.h

The parameter `HMMessageRecord` (a structure) is now passed as a pointer in the calls `HMShowBalloon`, `HMBalloonRect`, and `HMBalloonPict`.

DatabaseAccess.h

The pointer to the structure `ResListElem` has been changed from `ResListElemPtr` to `ResListPtr` in order to be consistent with the Pascal equivalent.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

Fcntl.h

Definitions for the following "internal-use only" macros have been removed from the header file FCntl.h.

```
O_TMP
O_USEP
```

Files.h

The function name `rstfLock` has been changed to `rstflock`.

IoCtl.h

The definition of the internal struct type `_SeekType` has been removed from IOCtl.h.

Notification.h

The field `nmsIcon` in the structure `NMRec` has been changed to `nmIcon`.

Numerics headers

Math.h and SANE.h have been modified to bring the content of Math.h closer to the specifications of the ANSI draft standard. Specifically, the declarations of the following functions have been moved from SANE.h to Math.h: `sqrt`, `log`, `exp`, `tan`, `cos`, `sin`, and `atan`. Also, Math.h defines the extended value HUGE_VAL as positive infinity. SANE.h now contains a *#include* of Math.h. Formerly it was the other way around: Math.h included SANE.h.

The result of these changes is that Math.h now provides the user with an interface to just that mathematical functionality provided by the ANSI C draft standard. SANE.h extends the interface to the functionality provided by the Standard Apple Numerics Environment.

Implementations of the numerical functions declared in Math.h and SANE.h continue to reside in the same library files as hitherto. By linking to both CSaneLib.o and Math.o (or, if appropriate, CSaneLib881.o and Math881.o), the user is assured of obtaining all of the functionality described by both header files.

The `haltvector` prototype has been changed to take correct parameters.

Time.h

`#define CLK_TCK 60` has been changed to `#define CLOCKS_PER_SEC 60`.

The type cast in the `#define` for `difftime` has been changed to `long double`.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

Changes to Pascal Interfaces Only

IntEnv.p

The name of the variable `EnvP` has been changed to `_EnvP`.

Sound.p

The field `sampleArea` in `SoundHeader`, in `CmpSoundHeader`, and in `ExtSoundHeader` is now `PACKED ARRAY OF Byte`. This change was required in order to obtain the correct size allocation; if `ARRAY OF Byte` is used, two bytes are allocated for every member of the array.

The following have had their types changed. The field `dbSoundData` in the `RECORD SndDoubleBuffer` is now a `PACKED ARRAY OF Byte`. The field `count` in in the `RECORD LeftOverBlock` is now a `LONGINT`.

SoundInput.p

The type of the parameter `filterProc` in the functions `SndRecord` and `SndRecordToFile` has been changed from a `procPtr` to a `ModalFilterProcPtr` to be consistent with the equivalent C interface.

Changes to C and Pascal Interfaces

In the following, every instance of a file name shown in boldface denotes the C and Pascal equivalents. For example, **types** denotes types.h and types.p. Where applicable, these changes are also reflected in the assembly language files.

A number of error codes that were previously defined in the individual header files have been moved to **Errors**.

Synonyms have been added to permit replacement of the following undesirable spellings:

All calls beginning with "Dispos" can now alternatively begin with "Dispose". Similarly, "Updt" can be replaced by "Update". Note: MacsBug currently still reports the old version of these names.

AppleTalk

The Phase II AppleTalk interfaces have been reworked.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

Controls

References to `AuxCtlHndl` were changed to `AuxCtlHandle`.

EPPC

The constant name `rtrnReciptMsgID` has been corrected to `rtrnReceiptMsgID`.

Errors

Constants were added for debugger user breaks (-490, -491, -492).

Files

The following synonyms have been added for existing but obsolete calls: `GetForeignPrivs` for `GetAltAccess` and `SetForeignPrivs` for `SetAltAccess`.

In the structure `GetVolParmsInfoBuffer`, the type of `vMServerAdr` has been changed from `long` to `Handle`.

The calls `FSpCreateResFile` and `FSpOpenResFile` have been moved from **Files** to **Resources.**

The structures `FXInfo` and `DXInfo` have been changed to add the fields `fdXFlags` and `frXFlags`.

GestaltEqu

The following two constants (selectors) have been added: `gestaltHasFSSpecCalls` and `gestaltHasFileSystemManager`.

Icons

The constant `SpoolFolderIconResource` has been renamed `printMonitorFolderIconResource`.

Memory

The name `TempMem` has been added as a synonym for the obsolete call `MFTemp`.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

OSUtils

The following calls have been added: `GetOSTrapAddress`, `SetOSTrapAddress`, `GetToolboxTrapAddress`, `SetToolboxTrapAddress`,`GetToolTrapAddress`, and `SetToolTrapAddress`.

The constant `envPortADBKbd` has been changed to `envPrtblADBKbd`.

The constant `envPortISOADBKbd` has been changed to `envPrtblISOKbd`.

International keyboard equates were changed from `xxxIntlxxx` to `xxxISOxxx`.

Palettes

The interface has been updated for 32-bit Quickdraw.

The return type of `HasDepth` has been changed to `short`.

PictUtil

The following constants have been renamed: `METHOD_SYSTEM` has been changed to `systemMethod`, `METHOD_POPULAR` has been changed to `popularMethod`, and `METHOD_MEDIAN` has been changed to `medianMethod`.

Power

The Power Manager interfaces have been completely reworked.

PPCToolBox

The constant `loginCancelErr` has been changed to `userCancelledErr`. In the functions `GetDefaultUser` and `StartSecureSession`, occurrences of the parameter types `Str255` have been replaced by `Str32`.

QDOffscreen

The constant `useMFTempBit` has been changed to `UseTempMemBit`.

Quickdraw

The interface has been updated for 32-bit Quickdraw.

The type name `PICT2Header` has been changed to `OpenCPicParams`.

`ColorCompareProcPtr` has been changed to `ColorComplementProcPtr`.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

Sane

The `mischaltinfo` data structure has been added.

The `X80t9X96` function has been removed; the function never existed.

Script

The constants `ring` and `breve` have had their names changed, respectively, to `ringMark` and `breveMark`. The correctly spelled constant, `tokenCaret`, has been added; the incorrectly spelled version, `tokenCarat` has been retained. The unpublished KeyScript "verb" `smKeyModalDialog` has been removed.

Slots

The field `spParamData` in the structure `SpBlock` has been changed from `Ptr` to `long`; this corrects a long-standing error.

Sound

The Sound Manager interfaces have been completely reworked. Specific changes follow:

`SndGetBuffer` and `SndGetMixerLoad` have been removed.

The following constants have had their names changed: `noteSynth` has been changed to `squareWaveSynth`, `noteCmd` has been changed to `freqDurationCmd`, `baseNote` has been changed to `baseFrequency`, `numSampleFrames` has been changed to `numFrames`, and `dataPointerFlag` has been changed to `dataOffsetFlag`.

SoundInput

The following constant has been removed: `siRTFOnOff`. The following constants have been added: `siContinuous`, `siActiveChannels`, and `siActiveLevels`.

Traps

The constant `_UnmountVol` has been changed to `_UnMountVol`.

Types

The type `Str32` has been moved from **AppleTalk** to **Types.**

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

The following `Ptr` types have been added: `FixedPtr`, `FractPtr`, `OSTypePtr`, `ResTypePtr`, `PointPtr`, and `RectPtr`. For Pascal only, `IntegerPtr` and `LongIntPtr` have been added.

Windows

Reference to AuxWindHndl were corrected to read AuxWinHandle.

Changes to Assembler Interfaces

The files HardwareEqu.a and Private.a are no longer supported.

PaletteEqu.a

The macros `_Entry2Index`, `_RestoreDeviceClut`, and `_ResizePalette` have been moved to Traps.a.

PPCToolbox.a

This interface file has been completely rewritten.

Processes.a

The AppleEvent message types for `kSystemProcess` have been deleted.

- Note: Developers should check the AppleEvent registry for AppleEvent type declarations.

QuickEqu.a

The following have been removed as obsolete and unpublished spellings:

`QDSMgrSlop`, `QDSMgrCharExtra`, `rgnOverFlowErr`, and `insufficientStackErr`.

Instead of `rgnOverFlowErr`, use `rgnTooBigError` in SysErr.a; instead of `insufficientStackErr`, use `nsStackErr` in SysErr.a; instead of `QDSMgrSlop`, use `qdRunSlop` in QuickEqu.a; instead of `QDSMgrCharExtra`, use `qdChExtra` in QuickEqu.a.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

SANEMacs.a

An omission has been corrected by the addition of the macro `FCLASSC` to SANEMacs.a. (See *Apple Numerics Manual, Second Edition*, pages 153 and 273.)

ScriptEqu.a

The calls `sisHighCall` and `fisHighCall` have been removed.

In the structure `NItl4Rec`, the field `resHeader1` has had its name changed to `format`.

StandardFile.a

The spelling of `Goto` has been changed to `GoTo`. A typo was corrected, which changes `CustomPGetFile` to `CustomGetFile`.

SysEqu.a

The following constants have been changed: `envPortADBKbd` to `envPrtblADBKbd` and `envPortISOADBKbd` to `envPrtblISOKbd`.

TimeEqu.a

All "private" equates were removed and placed in Private.a.

ToolEqu.a

All occurrences of `Styl` have been changed to `Style`.

Traps.a

The macro `QDOffscreen` has been changed to `QDExtensions`.

The spelling of `Pixmap32Bit` has been changed to `PixMap32Bit`.

Changes to RIncludes

`Country` has been changed to `Region` (SysTypes.r).

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

The color resource type definitions in Types.r have been given predefined values to simplify their use. The old type definitions are still present in Types.r and can be used by defining the variable `oldTemp`, e.g. on the Rez command line. Existing resources can be converted to work with the new type definitions by using DeRez on them with `oldTemp` defined, deleting all data items that correspond to predefined values, and then using Rez with the new definitions, i.e. with `oldTemp` undefined.

In BalloonTypes.r all occurrences of `hmpic` have been changed to `hmpict`.

Operating System Calls

Deleted Calls

The call `Restart`, formerly declared in OSUtils.h and OSUtils.p has been removed. The facilities of the Shutdown Manager should be used instead. Ref.: Inside Macintosh, Vol. V, Ch. 32.

The calls `RAMSDOpen` and `RAMSDClose`, formerly declared in Serial.h and Serial.p have been removed. See Macintosh Technical Note No. 249 for information on opening the Serial Driver.

Synchronous/Asynchronous

New function calls have been added corresponding to many functions that have as a parameter the ASync/Sync flag. For function `<OSCall>`, the new functions would be named `<OSCall>Sync` and `<OSCall>ASync`. E.g., the calls corresponding to `PBOpen` would be `PBOpenSync` and `PBOpenASync`. These have all been implemented as inlines, which increases speed and decreases code size.

Known Outstanding Bugs

The item listed below is a bug that existed in MPW 3.0.

The macro for the Disk Initialization Package in PackMacs.a and some of the new System 7.0 calls in Traps.a which use the `DoDispatch` macro will not work with the assembler when using the `CASE ON` directive. This is of particular interest to MacApp users because the MacApp default for the assembler is set to be case sensitive.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

Bug Fixes

The items listed below are bugs that existed in MPW 3.0 or MPW 3.1. The list does not include bugs that first appeared in Alpha or Beta versions of MPW 3.2.

- In various function calls, pointers to `AddrBlock` were replaced by instances of `AddrBlock`.
- In Files.p and Files.h, `HParamBlkPtr` was changed to `ParamBlkPtr` for `PBSetVInfo`.
- In Printing.h, the `PItemProcPtr` typedef was fixed to return a `void`.
- In Quickdraw.p and Quickdraw.h:

 The old-format color constants were corrected.

 The `InsetRect` call was corrected to take a `VAR rect`.

 The `QDProcsPtr` was changed to `CQDProcPtr` in the `CQDProcs` record.
- In Resources.h, the type of `LoadResource` was changed from `void` to `pascal void`.
- In SCSI.p, the arrays in `PartitionRecord` were changed to packed arrays.
- In Sound.p and Sound.h, the typdefs for names of the form `Snd...ProcPtr` were changed from returning `Boolean` to returning `void`.
- In Traps.p and Traps.h, the trap number for `WriteParam` trap was corrected.
- In PackMacs.a, the case of various routine selector constants was changed to match the case used in the calls.
- In ROMEqu.a, the `osLstEntry` macro was corrected to mask the low 24 bits of the offset parameter.
- In ObjMacros.a, several bugs in `ObjectTemplate` macros and a bug in calling inherited methods were fixed.
- In SysEqu.a, an erroneous `EQU` for `BNMQHd` was corrected.
- In SysErr.a, `hwParamrErr` was corrected to `hwParamErr`.

Copyright Apple Computer, Inc. 1990-1991. All rights reserved.

MPW 3.2 Simple Input/Output Window
Release Notes

The Simple Input/Output Window (SIOW) is a package that creates a special kind of application (see "Restrictions" below). It was developed in order to enable C and Pascal programs that were not originally written for the Macintosh to exhibit, at least partially, typical Macintosh appearance and behavior. It also reduces the number of code changes needed for porting them in comparison to porting them to MPW. The programs built with this facility can be launched in the same manner as an application, i.e. by double-clicking them or by selecting them and performing a menu or command-key *open* command. Input/Output interactions with the program take place in a window. The input behavior differs from that of MPW in two respects: an input operation is terminated by either the "return" key or "enter" key; if a prompt has been output, the prompt will not usually be considered to be part of the input that follows, even if both are on the same line.

Creating an SIOW Application

The SIOW folder contains three items: a library (SIOW.o), a resource file (SIOW.r), and a header (SIOW.h) that is used by the resource file. Assuming that a build script already exists for your program as a tool, the modifications needed to build it as an SIOW application are the following:

1. Add the library `{MPW}:SIOW:SIOW.o` to your link prior to `{MPW}Runtime.o`.
2. Change the `-t` option of the link to `'APPL'`.
3. Perform the command `rez -a {MPW}:SIOW:SIOW.r` to your object file.

The linker should give warnings with respect to duplicate symbol definitions for `_coExit`, `_coWrite`, and `_coRead`.

An alternative procedure is to build using CreateMake, selecting the SIOW radio button. If CreateMake is used, the link command line will contain the `-d` option; therefore the duplicate symbol warnings mentioned above will be suppressed. If you wish, when using this alternative procedure, to be warned about (possibly other) duplicate symbol definitions, you should edit the makefile to remove this option.

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

Operation

The SIOW library intercepts any read or write calls to the console driver. It will only recognize calls to stdin, stdout, and stderr (in Pascal, respectively: input, output, and diagnostic). Any I/O to files or to other windows is unaffected.

Whenever the console driver is called, a window is created and brought to the front, complete with a menu bar with the menus: File, Edit, Font, Size, and the Apple menu. This window has a *zoom box*, a *grow box*, vertical and horizontal scroll bars, and bears the name "Untitled" until renamed by use of *Save* or *Save As*.

Menu Items

Apple Menu

The *Apple* menu contains the usual items: About SIOW, DA's, and multifinder list. The SIOW AboutBox gives copyright information.

File Menu

The *File* menu contains the items Save As, Save, Page Setup, Print, and Quit. The items Open, New, and Close are currently disabled (dimmed); they remain in the menu for future use.

Save As...

This will save the contents of the window as a file of type `'TEXT'` with creator `'????'`. A dialogue box will appear asking the user to enter the name of the new file. The saved file can be opened from within MPW. Double clicking on it from the finder will elicit the "Application not found" error because the creator was not set for the file.

Save

This will save the contents of a previously saved window, provided changes have occurred since the previous save. If the window has not been previously saved, this command operates as "Save As...".

Page Setup

This presents the usual dialogue box for setting printing parameters, e.g. page orientation.

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

Print

This will print the window with the name of the window and the word "page" followed by the page number at the foot of each printed page.

Quit

This will quit the SIOW program via ExitToShell. If the current window contents have not been saved, a dialogue box will appear to ask if it should be saved. The user is given three options: Yes, No, or Cancel. The user can click the corresponding buttons, or enter respectively "y", "n", or "command-period" from the keyboard.

Edit Menu

This menu contains Undo, Cut, Paste, and Copy. They behave in the usual manner.

Font Menu

The font menu contains all fonts available in your system. The default is Monaco.

Size Menu

This contains the size list: 9, 10, 12, 14, and 18. The sizes that look best with the selected font are shown in an outline style. The default size is 9 point.

Output

Initial output is at the top left of the window. Subsequent output will appear at the current location of the insertion point.

The user does not need to worry about I/O buffering when doing ordinary interactive sequences of alternating output and input, because flushing of the output buffer is automatic before execution of an input statement and before program exit. However, the user should be aware of the need to flush buffers when writing output statements for debugging purposes in order to be sure of seeing such output before a program crash. The things to be remembered are that *stderr* (*diagnostic* in Pascal) is line buffered whereas *stdout* (*output* in Pascal) is fully buffered. Therefore, to ensure obtaining debugging output, one should do one of the following:

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

C

Either output to *stderr*, making sure that the output ends with "\n", or include `stdio.h` in your source and follow each output to *stdout* with the call `fflush(stdout)`.

Pascal

Either output to *diagnostic* using Writeln, or put a `USES PasLibIntf` in your source and follow each output to *output* with the call `PLFlush(output)`.

Input

If, following receipt of output, characters are typed and either *return* or *enter* are pressed, only the typed characters are read. If the insertion point is moved by the mouse or the arrow keys, then the entire line that contains the insertion point will be read. If material is selected, so that it appears highlighted, then all of the selected material will be read.

- *Note:* The input behavior differs from that found in MPW. In the latter, an output followed by an input, all on the same line, will unconditionally result in reading the entire line.

Known Outstanding Bugs

- Certain I/O errors, encountered while saving a window, will not be reported correctly. The error message presented in the alert box will be incorrect, missing, or garbled.
- Using the "Undo" menu command after deleting text in the window will cause some of the remaining text in the window to be overwritten. Overwriting starts at the current insertion point or start of the current selection, and the number of bytes overwritten is equal to the number that had been previously deleted. The manner in which the deletion occurs is immaterial; it can be delete, or "Cut," or "Clear."
- If the cursor is repositioned in the edit window for the purpose of sending a line of text to the program as input, an extra carriage return will be returned in the input stream. That is, the input line will be terminated with two consecutive carriage returns. This can cause the next input request to begin with a *newline*, which can cause a null input line to be read (e.g., in the case of input from `gets()` in C or `READLN` in Pascal).

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

- When printing an SIOW document, the text will overlap the SIOW-generated footer if there is a full page of text.

Restrictions

SIOW was designed to run in any language that was linked with the MPW libraries. The library I/O design allows interception of any low level read or write call to the console. This behavior can be guaranteed for any procedural language.

SIOW is not recommended for programs that are considered true Macintosh applications. If your program already brings up windows, and handles I/O, avoid this package.

The SIOW application requires a minimum of 38K of memory. It will more comfortably run at 122K. Any application made with SIOW will run on Macintoshes with 128K (or larger) ROMS.

Copyright Apple Computer, Inc.
1990-1991. All rights reserved.

APPLE COMPUTER, INC. SOFTWARE LICENSE

PLEASE READ THIS LICENSE CAREFULLY BEFORE USING THE SOFTWARE. BY USING THE SOFTWARE, YOU ARE AGREEING TO BE BOUND BY THE TERMS OF THIS LICENSE. IF YOU DO NOT AGREE TO THE TERMS OF THIS LICENSE, PROMPTLY RETURN THE UNUSED SOFTWARE TO THE PLACE WHERE YOU OBTAINED IT AND YOUR MONEY WILL BE REFUNDED.

1. **License.** The application, demonstration, system and other software accompanying this License, whether on disk, in read only memory, or on any other media (the "Apple Software") and related documentation are licensed to you by Apple. You own the disk on which the Apple Software is recorded but Apple and/or Apple's Licensor(s) retain title to the Apple Software and related documentation. This License allows you to use the Apple Software on a single Apple computer and make one copy of the Apple Software in machine-readable form for backup purposes only. You must reproduce on such copy the Apple copyright notice and any other proprietary legends that were on the original copy of the Apple Software. You may also transfer all your license rights in the Apple Software, the backup copy of the Apple Software, the related documentation and a copy of this License to another party, provided the other party reads and agrees to accept the terms and conditions of this License.

2. **Restrictions.** The Apple Software contains copyrighted material, trade secrets and other proprietary material and in order to protect them you may not decompile, reverse engineer, disassemble or otherwise reduce the Apple Software to a human-perceivable form. You may not modify, network, rent, lease, loan, distribute or create derivative works based upon the Apple Software in whole or in part. You may not electronically transmit the Apple Software from one computer to another or over a network.

3. **Support.** You acknowledge and agree that Apple may not offer any technical support in the use of the Software.

4. **Termination.** This License is effective until terminated. You may terminate this License at any time by destroying the Apple Software and related documentation and all copies thereof. This License will terminate immediately without notice from Apple if you fail to comply with any provision of this License. Upon termination you must destroy the Apple Software and related documentation and all copies thereof.

5. **Export Law Assurances.** You agree and certify that neither the Apple Software nor any other technical data received from Apple, nor the direct product thereof, will be exported outside the United States except as authorized and as permitted by the laws and regulations of the United States.

6. **Government End Users.** If you are acquiring the Apple Software on behalf of any unit or agency of the United States Government, the following provisions apply. The Government agrees:

(i) if the Apple Software is supplied to the Department of Defense (DoD), the Apple Software is classified as "Commercial Computer Software" and the Government is acquiring only "restricted rights" in the Apple Software and its documentation as that term is defined in Clause 252.227-7013(c)(1) of the DFARS; and

(ii) if the Apple Software is supplied to any unit or agency of the United States Government other than DoD, the Government's rights in the Apple Software and its documentation will be as defined in Clause 52.227-19(c)(2) of the FAR or, in the case of NASA, in Clause 18-52.227-86(d) of the NASA Supplement to the FAR.

7. **Limited Warranty on Media.** Apple warrants the disks on which the Apple Software is recorded to be free from defects in materials and workmanship under normal use for a period of ninety (90) days from the date of purchase as evidenced by a copy of the receipt. Apple's entire liability and your exclusive remedy will be replacement of the disk not meeting Apple's limited warranty and which is returned to Apple or an Apple authorized representative with a copy of the receipt. Apple will have no responsibility to replace a disk damaged by accident, abuse or misapplication. ANY IMPLIED WARRANTIES ON THE DISKS, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO NINETY (90) DAYS FROM THE DATE OF DELIVERY. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

8. **Disclaimer of Warranty on Apple Software.** You expressly acknowledge and agree that use of the Apple Software is at your sole risk. The Apple Software and related documentation are provided "AS IS" and without warranty of any kind and Apple and Apple's Licensor(s) (for the purposes of provisions 8 and 9, Apple and Apple's Licensor(s) shall be collectively referred to as "Apple") EXPRESSLY DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. APPLE DOES NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE APPLE SOFTWARE WILL MEET YOUR REQUIREMENTS, OR THAT THE OPERATION OF THE APPLE SOFTWARE WILL BE UNINTERRUPTED OR ERROR-FREE, OR THAT DEFECTS IN THE APPLE SOFTWARE WILL BE CORRECTED. FURTHERMORE, APPLE DOES NOT WARRANT OR MAKE ANY REPRESENTATIONS REGARDING THE USE OR THE RESULTS OF THE USE OF THE APPLE SOFTWARE OR RELATED DOCUMENTATION IN TERMS OF THEIR CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE. NO ORAL OR WRITTEN INFORMATION OR ADVICE GIVEN BY APPLE OR AN APPLE AUTHORIZED REPRESENTATIVE SHALL CREATE A WARRANTY OR IN ANY WAY INCREASE THE SCOPE OF THIS WARRANTY. SHOULD THE APPLE SOFTWARE PROVE DEFECTIVE, YOU (AND NOT APPLE OR AN APPLE AUTHORIZED REPRESENTATIVE) ASSUME THE ENTIRE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO YOU.

9. **Limitation of Liability.** UNDER NO CIRCUMSTANCES INCLUDING NEGLIGENCE, SHALL APPLE BE LIABLE FOR ANY INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES THAT RESULT FROM THE USE OR INABILITY TO USE THE APPLE SOFTWARE OR RELATED DOCUMENTATION, EVEN IF APPLE OR AN APPLE AUTHORIZED REPRESENTATIVE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME STATES DO NOT ALLOW THE LIMITATION OR EXCLUSION OF LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

In no event shall Apple's total liability to you for all damages, losses, and causes of action (whether in contract, tort (including negligence) or otherwise) exceed the amount paid by you for the Apple Software.

10. **Controlling Law and Severability.** This License shall be governed by and construed in accordance with the laws of the United States and the State of California, as applied to agreements entered into and to be performed entirely within California between California residents. If for any reason a court of competent jurisdiction finds any provision of this License, or portion thereof, to be unenforceable, that provision of the License shall be enforced to the maximum extent permissible so as to effect the intent of the parties, and the remainder of this License shall continue in full force and effect.

11. **Complete Agreement.** This License constitutes the entire agreement between the parties with respect to the use of the Apple Software and related documentation, and supersedes all prior or contemporaneous understandings or agreements, written or oral, regarding such subject matter. No amendment to or modification of this License will be binding unless in writing and signed by a duly authorized representative of Apple.

7/15/91
001-0158-A

M0615LL/B

Macintosh® Programmer's Workshop Toolbox Interfaces and Libraries

Version 7.0

This package contains

1	Set of release notes	*MPW® 7.0 Toolbox Interfaces and Libraries Release Notes*
5	Disks	*MPW Toolbox Interfaces and Libraries 1,* Disk 1 of 5 *MPW Toolbox Interfaces and Libraries 2,* Disk 2 of 5 *MPW Toolbox Interfaces and Libraries 3,* Disk 3 of 5 *MPW Toolbox Interfaces and Libraries 4,* Disk 4 of 5 *MPW Toolbox Interfaces and Libraries 5,* Disk 5 of 5

If you have any questions, please call

1-800-282-2732 (U.S.)
1-408-562-3910 (International)
1-800-637-0029 (Canada)

4/29/91